The Report Committee for Andrew John Evans
Certifies that this is the approved version of the following report:

Design, Implementation, and Measurements of a High Speed Serial Link Equalizer

APPROVED BY
SUPERVISING COMMITTEE:

Supervisor:

Adnan Aziz

Mark McDermott

Design, Implementation, and Measurements of a High Speed Serial Link Equalizer

by

Andrew John Evans, BSCPE

Report

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN ENGINEERING

The University of Texas at Austin

December 2012

In memory of Bernard Seger and Stephen Kyle McNulty

Acknowledgements

I would like to acknowledge all of the educators who have influenced my life. Without their steadfast dedication to teaching I would not be where I am today. Specifically, I would like to thank Adnan Aziz and Mark McDermott for supervising and reading this report. In addition to academic guidance I would like to acknowledge the personal support and motivation I have received from my family and friends. My parents, John and Kathy, and late grandfather, Bernard, have always provided a foundation for my continual pursuit of knowledge. I would like to thank all of my friends and colleagues for providing constant moral support throughout my time spent in this program. Finally, I would like to thank my first childhood friend, Stephen, for helping me draw strength from within in times of adversity.

Design, Implementation, and Measurements of a High Speed Serial Link Equalizer

Andrew John Evans, M.S.E

The University of Texas at Austin, 2012

Supervisor: Adnan Aziz

The advancements of semiconductor processing technology have led to the ability for computing platforms to operate on large amounts of data at very high clock speeds. To fully utilize this processing power the components must have data continually available for operation upon and transport to other system components. To enable this data requirement, high speed serial links have replaced slower parallel communication protocols. Serial interfaces inherently require fewer signals for communication and thus reduce the device pin count, area and cost. A serial communication interface can also be run at a higher frequency because the clock skew between channels is no longer an issue since the data transmitted on various channels is independent. Serial data transmission also comes with a set of drawbacks when signal integrity is considered. The data must propagate through a channel that induces unwanted effects onto the signals such as intersymbol interference. These channel effects must be understood and mitigated to successfully transmit data without creating bit errors upon reception at the target component. Previously developed adaptive equalization techniques have been used to filter the effects of intersymbol interference from the transmitted data in the signal. This report explores the modeling and implementation of a system comprised of a transmitter, channel, and receiver to understand how intersymbol interference can be removed

through a decision-feedback equalizer realized in hardware. The equalizer design, implementation, and measurements are the main focus of this report and are based on previous works in the areas of integrated circuit testing, channel modeling, and equalizer design. Simulation results from a system modeled in Simulink are compared against the results from a hardware model implemented with an FPGA, analog to digital converter and discrete circuit elements. In both the software and hardware models, bit errors were eliminated for certain amounts of intersymbol interference when a receiver with decision-feedback equalization was used instead of a receiver without equalization.

Table of Contents

List of Figures

Chapter 1:

High Speed Serial Communication and Intersymbol Interference

Modern computing platforms require high bandwidth communication interfaces between devices in the system to fully utilize the performance delivered by state-of-the-art micro-processor architectures and circuit technologies. Many of the older parallel communication protocols such as Parallel ATA, Small Computer System Interface (SCSI), and Industry Standard Architecture (ISA) have been replaced by higher speed serial interfaces for several reasons. Serial interfaces require fewer pins which reduces the area and cost of an integrated circuit. Since the clock skew between channels is not a major concern, serial links can also run at higher clock rates. Additionally, since there are fewer conductors in serial data links it is easier to isolate them from channel effects than it is with a parallel scheme requiring more wires.

Though most high bandwidth communication protocols are implemented serially to overcome the shortfalls of parallel communication, there are still major adversities to overcome. These issues must be mitigated to maintain the advantages that serial interfaces provide. Since the serial communication protocols run at a much higher clock rate the effects of intersymbol interference (ISI) [1] and crosstalk [2] are more pronounced on these signals. If these undesired channel effects induce enough distortion onto a signal then bit errors can occur and may outweigh the benefits of running at higher clock rates. There are several ways to reduce unwanted channel effects: encoded signaling schemes, increasing the magnitude of high frequency signal components relative to lower frequency ones, error-correcting code algorithms, and equalization schemes. None of these techniques provide a comprehensive solution to eliminate all of

the channel effects. For instance, error correcting codes add information redundancy to a signal to allow the detection and correction of a pre-defined amount of bit errors [3]. However, hardware and data cost of implementing error correcting codes becomes increasingly large as the amount of bit error correction rises. In comparison, equalization techniques can be used to reduce a large amount of bit errors for a certain type of channel with a relatively small amount of hardware and no data overhead. This report will focus on using equalization to filter ISI and reduce the amount of received bit errors.

A high speed serial link system model was simulated in software and implemented in hardware to explore the effects of channel induced ISI upon a serial link signal. A decision-feedback equalizer (DFE) hardware architecture was used to filter ISI and thus reduce bit errors that would be blindly detected by a receiver without equalization. The details of the transmitter, channel, and receiver models will be provided to set the context for application to any type of generic high speed serial link system. The implementation and tuning of the equalization hardware will be discussed to understand how particular circuit parameters affect the overall receiver performance. The results of both the software simulation and hardware implementation will be presented to explore the similarities and differences between both of the models.

1.1 INTRODUCTION

This report further explores the concepts of high speed serial link communication, adverse channel effects on signals and equalization architectures. It provides implementation details and experimental results of software and hardware models of an electronic transceiver sending serial data through a channel and receiving the data with

ISI induced upon it. The receiver data, collected both with and without equalization, is compared to the transmitted data for validation of correct data reception. In this chapter a high level description is provided to lay the foundation for the overall experiment. The problem statement is introduced to narrow the focus of the report and describe the key parameters that are analyzed in the results. Many of the concepts in this report were leveraged from prior academic and industry findings which will be briefly discussed in this chapter as well.

1.2 HIGH LEVEL DESCRIPTION

There are three main components of the system modeled and tested in this report: the transmitter, the channel and the receiver. The transmitter is the component which creates serial, binary data to send to another device in the system. In an industry standard communication protocol, such as Peripheral Component Interconnect Express (PCIe), the serial data may be encoded and sent in packets to deliver messages to other devices in a system. These data packets could contain information about the pixels displayed on a computer monitor, a recipe from an Internet website or even secure items such as ones bank account information. Whatever the information may be, the transmitter prepares the data and sends it out of the device using wired electrical signals (for the purpose of this report).

The channel is the medium through which the transmitted information propagates. There are many different types of communication channels in reality. A cellular phone sends data through the air and thus the channel for this information is the air itself. A boat mapping the topology of the ocean floor uses sonar to identify underwater structures.

This information is transmitted through water which acts as the channel in this case. For the scope of this report the channel is most simply defined as metal wires which carry information in the form of electrical signals. In an ideal world the channel would be able to deliver the electrical signal from the transmitter to the receiver instantly and without any information loss, i.e., the exact signal sent would be the exact signal received. However, due to the effects of the physical world there is always an inherent transmission delay and some amount of information loss between the transmitter and receiver due to the limitations of channel capacity [4].

Finally, the receiver is the component that recovers the information sent from the transmitter. The data has traveled through a channel that induces some amount of loss and disturbance upon the signal. It is a function of the receiver to accept and decode this information correctly. Imagine trying to pay a bill online but one of your identification numbers had changed due to the effects of the transmission channel. Without a proper receiver architecture this situation could plausibly occur. For this project the electrical signal that is immediately detected at the receiver's front end is not necessarily the data that will be ultimately interpreted and used by the logic within the receiving device. This information is however used as an input to a feedback system that makes a decision as to what the actual value should be. The feedback or equalization system explored in this report is the DFE. The hardware equalizer uses information about previous data received in an attempt to mitigate unwanted effects introduced by the channel. All of these elements are designed to operate together and allow data to be successfully sent from one point in the system to another even if there are many adversities to overcome in the non-ideal channel.

1.3 PROBLEM STATEMENT

A high speed electrical signal experiences dispersion when transmitted through a bandwidth limited channel and is affected by crosstalk when routed closely to other high speed signals. The effects of these channel adversities reduce the peak amplitude of the signal and spread the pulse width beyond one bit period or more. Figure 1 provides a visual representation of the channel effects in the time domain. This phenomenon is referred to as ISI and causes the data eye to close at the receiver. The data eye is a representation of multiple data bits overlaid at the receiver input and provides insight into the overall performance of the system. In general, a data eye with a large opening means that the signal is synchronized well from one bit period to another and the channel effects are not distorting the signal to a point where it cannot be received properly. As the data eye begins to shrink this indicates that the signal is being affected by ISI to the extent where it may be difficult or impossible to recover the transmitted data.

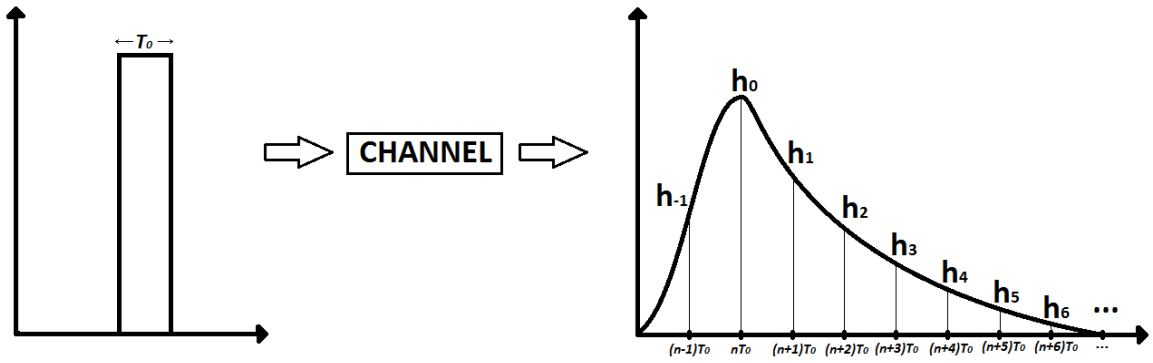

Figure 1: Pulse Signal Dispersion through a Channel

Unlike random noise, ISI is a deterministic property of the channel and equalization techniques exist to mitigate these effects. Specifically, a DFE uses previous bit values to estimate their effect on the current incoming bit to the receiver. Through feedback, the DFE is able to use this information from previous bits and filter it from the

current bit. The end user of an electronic platform may not even be aware of the challenges to overcome channel effects through equalization techniques. However, there would certainly be a significant impact to the user experience if these equalization schemes were not in place.

Consider an online e-commerce entrepreneur who uses his laptop computer to create websites that advertise his custom decanters online. He posts high-definition videos on the website to enhance the aesthetic experience and convey his selling points to consumers. He uses a video editing program to stitch together custom videos taken of his products as well as stock videos from the Internet to enhance the website's professional feel. Whether he knows it or not there is a state-of-the-art graphics processor on the laptop motherboard. This GPU communicates with a north bridge chip that acts like an IO traffic router between system memory, the CPU and GPU. The communication protocol for this network of system devices is a high-speed serial interface called PCIe. This protocol enables the main computation and communication components of his laptop to process and transport the high-definition video data he uses for the website.

The engineers that designed the computing components in this entrepreneur's laptop spent a great amount of time and effort to allow him to seamlessly transfer data at very high rates without error through these components. The chip design engineers were aware that routing high speed signals through extremely narrow (tens of nanometers) and closely routed metal wires would create crosstalk between them. Crosstalk is a physical phenomenon wherein the signal on one wire would affect the signal on another wire through capacitive effects even though they are not physically touching. They also knew that these narrow wires would not be able to carry all of the frequency component

information in this data because the energy in these signals could not be completely contained in the metal wires. All wires have a cutoff frequency at which the energy in a signal becomes attenuated. The cutoff frequency is determined by the dielectric properties of the material that the wires are composed of and the wires' impedance which is based off of its geometry and interactions with other materials close in proximity. To compensate for these non-ideal properties that manifest in physical reality, engineers developed equalization circuits to approximate these unwanted effects and compensate for them.

In a fully designed laptop system the processing, memory and communication components operate in tandem to compensate for these unseen yet important electrical effects. As computer technology continues to get faster and smaller it is important that things such as channel effects are compensated for by innovative circuit techniques when the boundaries of physics are being pushed. Without these advances in equalization technology the e-commerce entrepreneur's laptop would have to be the size of a small cabinet and as slow as a computer system from the late 1990s. Imagine lugging a system like that to your local coffee shop.

1.4 SUMMARY OF PRIOR WORK

High speed serial links and adaptive equalization schemes have been implemented and improved upon for quite some time. For instance in 2003, PCI-SIG introduced PCIe 1.0a, with a per-lane data rate of 250 MB/s and a transfer rate of 2.5 GT/s [5] (T/s refers to Transfers per second because this unit includes the overhead of the data encoding scheme). Also, the adaptive decision-feedback equalization scheme has had many papers

written about it dating back to the late 1960s [6] and early 1970s [7]. Reaching even further back, this report builds from the principle that any repeated signal may be viewed as being composed of an infinite amount of sinusoidal elements which has been discussed in the legendary works of Fourier. Using this principle Harry Nyquist was able to determine sufficient conditions to successfully transmit and receive a square wave signal even in the midst of distortion [8]. This is a fundamental principle used by the decision-feedback equalizer to realize that distortion on a data signal can be removed or compensated for if the characteristics of the distortion are known. Similar to all academic efforts of the mind, the work described in this report "stands on the shoulders of giants". The motivating factors for exploring the concepts discussed in this report are to combine several theories, models and applications to implement a software and hardware design that tackles modern day problems. The main tenants of prior work that were referenced to perform the experiments described in this report focus on test pattern generation, channel modeling and DFE implementation.

One of the fundamental concepts used in this report is the generation of pseudo-random bit sequences for testing hardware circuits. This is implemented through the use of linear feedback shift registers (LFSR) which have properties to facilitate pseudo-exhaustive test coverage [3] through common hardware elements. The internal structure of these circuits is described by viewing their characteristic polynomials of the form, $f(x) = 1 + h_1 x + h_2 x^2 + \cdots + h_{n-1} x^{n-1} + x^n$. If the period of the LFSR is $T = 2^n - 1$, and a primitive (irreducible) polynomial *p(x)* of degree *n* over a Galois field *GF(2)* is able to divide $1 + x^T$ but not $1 + x^i$ for any integer $i < T$, then that polynomial provides a maximal length period for the LFSR [9]. Fortunately, there are readily available look up tables in existence [10] which provide the primitive polynomials for these types of

circuits. Thus, once the concepts of the LFSR are understood it is relatively straightforward to design one in hardware by creating a basic shift register and inserting linear XOR operators in the appropriate tap position that correspond to a characteristic polynomial. An LFSR is used to created test pattern data for the system in this report.

The channel is another extremely critical aspect in high speed serial link systems. As the clock rates of high speed serial links continue to increase, signal integrity concerns and issues have come along with them. It is important to understand the implications of manufacturing CMOS integrated circuits at tiny, nanometer scale geometries and how the physical design effects high speed signaling. Accurate channel models have been developed to provide circuit designers a means to understand and simulate the circuits to withstand the effects of ever-shrinking circuit topologies. A fairly straightforward yet accurate model for on-chip crosstalk is modeled through a capacitive coupling to a driven line [2]. Figure 2 shows a circuit diagram of this model for on-chip crosstalk.

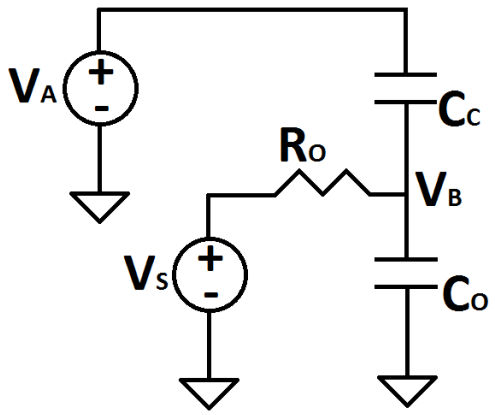

Figure 2: Capacitive Coupling to a Driven Capacitive Line

This is essentially a classical RC circuit, wherein a capacitive coupling coefficient is defined as $k_c = \frac{C_c}{C_0+C_c}$ and the time constant is defined as $\tau = R_0(C_C + C_O)$, where R_0

is the output resistance, C_C is the coupling capacitance and C_O is the output capacitance. The time domain response to a unit step voltage on a signal line coupled to another driven line is expressed by $V_B(t) = k_C e^{\frac{-t}{\tau}}$. This model is augmented further in the report but relies upon this fundamental concept to accurately design the experiment.

Finally, the most exciting and interesting concept implemented and explored in this report is the DFE. The primary focus of the experiments was to induce bit errors upon a signal traversing a non-ideal channel, implement DFE hardware to filter the channel effects, and receive the transmitted, equalized signal with reduced bit errors. Adaptive equalization techniques come in many varieties such as the least mean squares [11] or recursive least squares filters [12]. However, a common PCIe 3.0 receiver architecture implements a simple 2 tap DFE to overcome channel effects and transmit data at a rate of 8GT/s. Of course there are many other underlying factors at work that allow this protocol to operate at such high data rates but the simplicity and elegance of the DFE stand out. The basic topology of the DFE hardware circuit can be realized using a minimal amount of hardware [13]. For the purposes of this report the hardware implementation was kept fairly simple, i.e., it was not unrolled or pipelined. The general representation of a signal with ISI is given by the following equation.

$$r_n = a_n h_0 + (a_{n-1} h_1 + a_{n-2} h_2) + \sum_{i, i \neq 0,1,2} a_{n-i} h_i$$
$$= \text{Signal} + \text{Dominant ISI} + \text{Residual ISI}$$

In this expression, r_n is the received signal, a_{n-i} is the amplitude of the previous transmitted signal i periods previously, and h_i is the impulse function of a transmitted bit. The DFE implementation operates by computing a value equal to the dominant ISI and

subtracting it from the received signal before it is compared and decoded into a binary value. The theory and implementation of a DFE is concise and effective as are many of the notions explored in this report. "Simplicity is the ultimate sophistication." – *Leonardo da Vinci*

1.5 STRUCTURE OF REPORT

The remainder of this report provides an in depth exploration of the effects of ISI on high speed serial link communication, and the implementation of equalization techniques to minimize channel effects. Chapter 2 explicitly defines common terms and concepts used in this report so the reader is familiar with their particular usage in this context. Chapter 3 describes the models and test setups of the simulation and hardware experiments performed for this report. Chapter 4 analyzes the empirical data gathered throughout the experiments and provides a commentary to explain the results. Chapter 5 draws specific conclusions from the research and experimental results presented throughout the report, and suggestions will be made for future improvements to motivate the continual quest for knowledge and novel discoveries.

Chapter 2:

Formal Definitions

To provide a clear description of the experiments and ideas discussed in this paper a set of technical terms must be formally defined. This section defines these terms to provide a context for understanding the report.

The *transmitter* is the front end of the high speed serial link system. It converts binary data into electrical signals that are transmitted out of an integrated circuit. The binary data is sent through electrical drivers within the transmitter that supply current at the appropriate voltage levels for the signal to propagate through the channel. The output drivers typically consist of power amplifier circuit topologies to provide gain for weaker, internal logic signals. For the purpose of this experiment the transmitter resides in a field programmable gate array (FPGA) and drives the transmitted signal at voltage of 3.3V.

Intersymbol interference refers to the non-ideal channel effects that are induced upon a signal when a data bit is transmitted and interferes with subsequent transmitted bits. One can imagine ISI as the dispersion or blurring of bits from one period to another. A visual representation of ISI is shown in Figure 3:

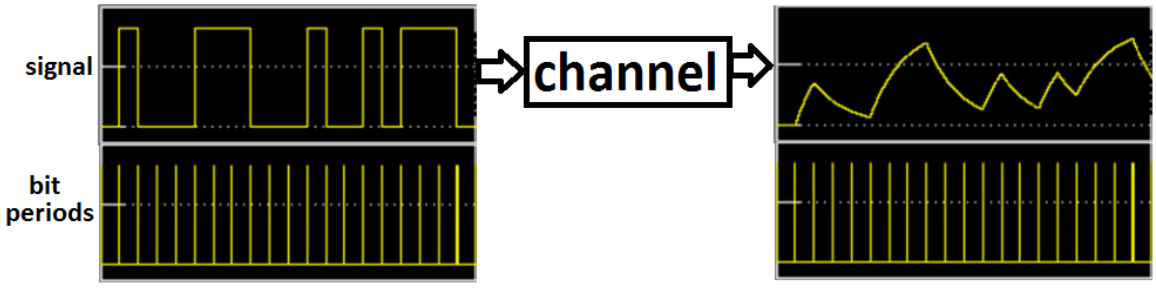

Figure 3: Intersymbol Interference through a Channel

The ideal signal which is transmitted can be seen on the left with an indication of the bit period transitions below it. This signal then propagates through a channel which introduces ISI upon the signal. When propagating through the channel, the ideal bit pulses are delayed in their rise and fall times, thus dispersing into other bit periods. For the context of this paper ISI will be considered only by the effects of capacitive crosstalk. The notion of random or thermal noise upon the signal will be discounted to provide a more streamlined focus on this particular type of ISI. This assumption can be made since the magnitude of random or thermal noise is much less than that of the effects of ISI in this environment.

Crosstalk refers to a type of noise induced by a particular signal to another signal through capacitive or inductive means. On-chip crosstalk is typically dominated by the capacitive coupling between signals. The geometry of the layout of these signals or the ability to tolerate a certain amount of crosstalk are really the only means of operating within its effects. When a voltage transition occurs on one signal this injects charge into adjacent signals and disturbs their voltage levels [2]. If the signal travels off chip it may be affected by the mutual inductance that occurs between interactions with other signals. For the purpose of this report only the capacitive effects will be considered to narrow the focus on the type of crosstalk that dominates on chip signaling.

The *channel* is the medium through which the electrical signal propagates. The channel is implemented by internal wires and external pins on the FPGA device, metal wires printed on a circuit board, a pin connector on the circuit board, wired connectors from the circuit board to a breadboard, and metal wires lined inside the breadboard. In addition to the inherent resistive, capacitive and inductive properties of the channel,

additional capacitance is explicitly added using discrete, ceramic capacitors. This capacitance models the ISI induced upon the signal by routing a target wire close to other aggressor wires. This type of routing would be done in a design using the latest circuit technology and highest speed serial link schemes such as PCIe 3.0.

The *analog to digital converter (ADC)* is a hardware component that accepts an analog electrical signal as an input and produces a digital representation of that signal as an output. The ADC is implemented as a 3rd party integrated circuit provided by Analog Devices, Inc. This particular ADC accepts the 3.3V electrical signal transmitted from the FPGA through the channel and produces an 8-bit digital representation of this signal as the output. To understand how the ADC output represents data for a given input, consider the following examples: a 0V analog input produces a digital output of 0x00, a 1.65V analog input produces a digital output of 0x80 and a 3.3V analog input produces a digital output of 0xFF. The ADC is a fundamental component required for the receiver equalizer to operate properly.

The *receiver*, located in the back end of the system, accepts electrical signals as inputs and converts them into binary, digital information stored within the integrated circuit. The receiver front end accepts a signal with a peak to peak voltage that is possibly larger than the voltages supported for on-chip, digital circuitry. Thus, the receiver converts the input voltage to levels that are optimal for the digital logic circuitry to operate. The receiver itself is responsible only for determining if an incoming electrical signal should be represented as a binary 0 or 1 internally. The analog receiver circuitry is inherently built into the FPGA pins.

The *decision-feedback equalizer (DFE)* is an adaptive equalization hardware architecture that uses voltage level values of previously received bits to provide a weighted sum feedback signal to cancel out the effects of ISI on the currently received data bit [14]. The DFE is implemented in the FPGA hardware and consists of registers, adders and shifters. The digital representation of the analog signal is used as the input to the DFE and is provided by the external ADC. The DFE uses this input value to perform adaptive equalization operations. Figure 4 shows the circuit architecture of the two tap DFE:

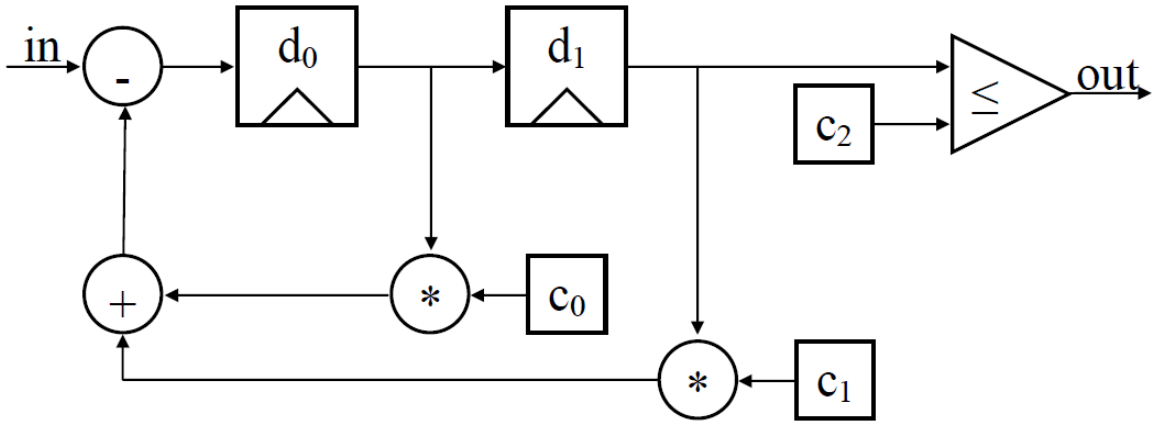

Figure 4: Decision Feedback Equalizer Circuit Architecture

The input *in* has a weighted sum value of previous decisions subtracted from it and is then stored in the memory element d_0. In the next clock cycle this value is multiplied by a constant value C_0 between 0 and 1 and is also stored again in memory element *d1*. In the clock cycle which is 2 clocks later the value is multiplied by a constant value C_1 between 0 and 1 and the two scaled values are summed to form the difference value. The output *out* is determined by comparing the output of d_1 to a

constant C_2 between 0x00 and 0xFF. The *out* signal is a binary representation of the received signal after equalization.

Bit errors occur when a transmitted binary data bit propagates through a channel and is received, with or without equalization, as the complement of the value that was transmitted. A standard measure of the amount of bit errors in a system is provided by the *bit error rate (BER)* metric. The BER is equal to the total amount of received bit errors divided by the total amount of bits transmitted over a particular period of time.

A *linear feedback shift register (LFSR)* is a shift register in which the input is a linear function of its previous state. The properties of this hardware enable pseudo-random data to be generated for a maximum of 2^N-1 periods, where N is the number of registers in the LFSR. Not all configurations of an LFSR result in the maximum amount of pseudo-random data generated. However, when the LFSR is configured properly and its period is 2^N-1 it is referred to as a maximal length LFSR because it will not hold the same state in the shift register until after 2^N-1 periods. Thus, a *pseudo-random bit sequence* (PRBS) can be generated by connecting a wire to one of the outputs of the registers in the LFSR [3]. A truly random bit sequence would transmit data each bit period that is completely independent of any other bit and is equally likely to be a 0 or 1. The LFSR acts as a pseudo-random data generator and has a deterministic output but exploits some of the properties of random data generators during its maximal length period of 2^N-1. For the purpose of this report a 20-bit LFSR was implemented and provided a maximal length period of 2^{20}-1 = 1,048,575 states. This LFSR generated a sufficient amount of sequences to allow the signal to charge and discharge the channel fully with strings of up to 19 bits of 0s and 1s. The characteristic polynomial of the 20-

bit LFSR is $x^{20} + x^{17} + 1$ [10]. To implement this equation in hardware, XOR functions were inserted in the positions defined by the characteristic polynomial. A visual representation of the LFSR used for this is shown in Figure 5:

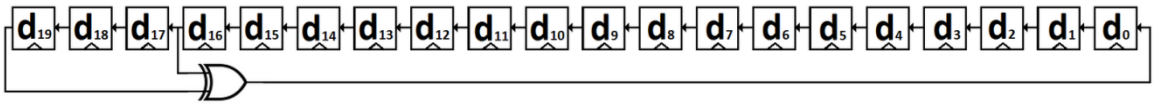

Figure 5: 20-bit Maximal Length Linear Feedback Shift Register

The LFSR was initially seeded with a value of 0x00001 which enabled the LFSR to operate properly and achieve its maximal length period. The output of register d_0 was tapped to use as an input to the transmitter. A tap in this context refers to the value on the wire of the input to the N^{th} register. In this scenario the LFSR output was used in a test setup, thus the entire LFSR was considered to be a *test pattern generator*.

Loopback refers to a type of transceiver test setup wherein a transmitter sends out a bit sequence that propagates through a channel and is decoded by the receiver. A passing test result occurs when the transmitted and received signals match. Often times the channel is desired to impose minimal effects on the transmitted signal. This allows for the functional behavior of the transmitter and receiver circuitry to be characterized instead of their ability to operate in an environment with significant channel effects. Once the transmitter and receiver are validated to operate functionally correct often times the test channel will intentionally induce ISI to test equalization circuitry within the receiver or transmitter [15]. The type of test setup described in this paper can be considered a form of loopback.

A hardware abstraction layer (HAL) is a software construct that allows higher level code to access hardware devices through a hardware independent interface [16]. Often times the HAL is located between the hardware device and the operating system, but is not considered to be part of the operating system. For the purpose of this report a HAL was developed to allow a high level C application to access FPGA hardware on a system level board. The HAL was coded in C and is registered with a Linux kernel running on an ARM9 processor.

Chapter 3:

Experimental Setup

To further explore serial high speed data link transmission and the ability for it to overcome ISI through equalization, both simulated and hardware systems were modeled to perform experiments exploring these concepts. The goal of the experiments was to induce ISI upon a known, pseudo-random signal and measure the BER of the received signal both with and without equalization. The parameters varied in the experiment were the amount and type of ISI induced upon the signal and the weighted tap parameters in the DFE. The following sections of the report describe the simulation and hardware setups.

3.1 SIMULATION

Modeling a system properly is the most important aspect of designing a specific, accurate experiment. The model must be explicit in defining the parameters of the various system components to focus on the parameters of interest in the experiment. Modern software packages provide powerful tools to perform both software and hardware modeling in the same environment. Simulink is a software package provided by Math Works that enables simulation and model based design in one convenient environment. For this experiment a model of the hardware setup was designed and simulated in Simulink to understand the system before the hardware was fully designed. The software model also provided a continuous environment for model refinement, test verification, and a baseline of data for comparison with hardware results. An additional benefit of using Simulink to model the design was its integration with Matlab, also provided by Math Works. The Matlab software package provides an extremely powerful

numerical computation engine and visualization packages through a high-level programming language. Using these two software tools in tandem, simulation data was generated by the Simulink model and a custom program was developed in Matlab to analyze the data. A visualization of the system modeled in Simulink is provided in Figure 6:

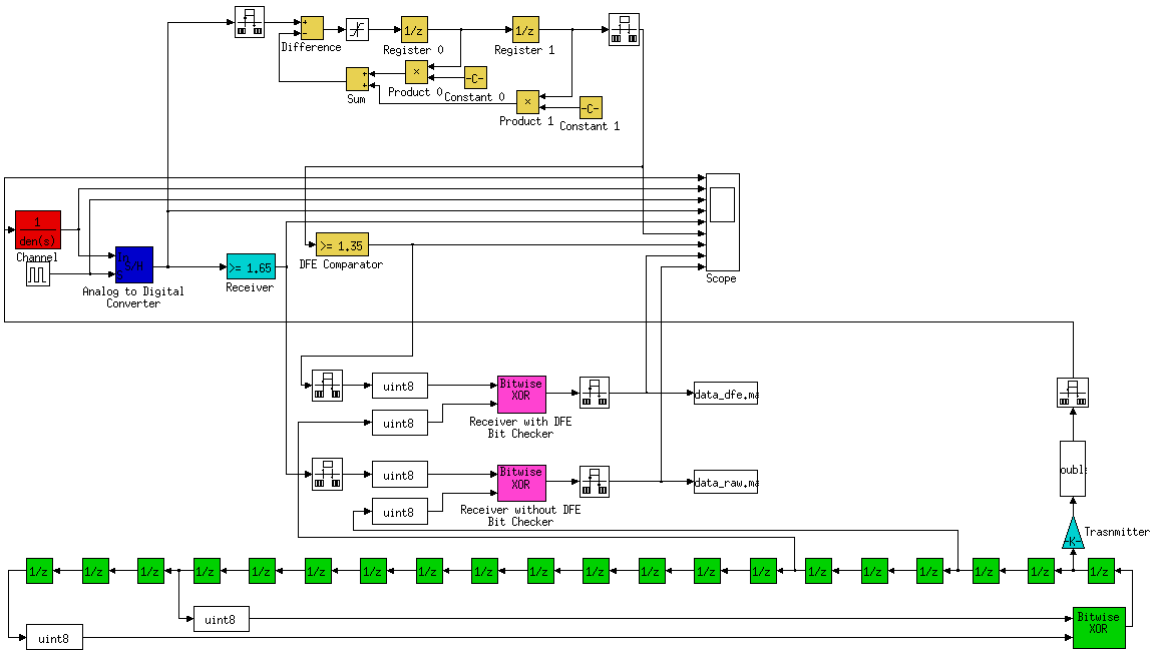

Figure 6: Simulink Model of High Sped Serial Link System with DFE

The first step in modeling the system was to create a known bit sequence that could be used as the "golden" transmitted data. The bit sequence had pseudo-random properties to exploit a balanced amount of 0s and 1s transmitted. This ensured that the receiver designed to compare voltages less than or equal to $\frac{VDD}{2}$ as 0 and greater than $\frac{VDD}{2}$ as 1 had an equal amount of opportunities to compare transmitted 0s and 1s. The known, pseudo-random bit sequence had a period greater than 1,000,000 before it repeated the same sequence. This ensured that strings of 0s and 1s up to 19 bits long propagated

through the channel. This causes the signal wire to charge and discharge completely at certain points throughout the sequence. To realize a known sequence with the properties described above a 20-bit LFSR was implemented in the Simulink model. The maximal length polynomial for the 20-bit LFSR was $x^{20} + x^{17} + 1$ and the initial seed was loaded as 0x00001. Notice that the linear XOR operators were placed on the 20^{th} and 17^{th} order taps and shifted back into the 0^{th} order tap. The green colored blocks in Figure 6 signify the Simulink model of the 20-bit LFSR which resided on the FPGA in the hardware model.

The LFSR produced binary values with no sense of scale or units attached to them. When the LFSR was realized in hardware this was also true. From the perspective of the system outside of the FPGA the values were manifest as external voltages propagating through the channel. To provide a sense of scale and units to the binary values of the LFSR output, a "gain" element was used in the Simulink model to set the peak to peak scale of the PRBS to 3.3. The triangular, cyan colored transmitter module is shown in Figure 6 above. This scale was used to model the 3.3V supply that the FPGA transmitter used to drive values out of the chip and through the channel. Though there was no sense of units in the Simulink model, it was sufficient to use the gain element since the other numerical operators in the system were agnostic to units as well.

The next module in the system to model was the channel. There are many ways to model a channel depending on its properties. To focus on the effects of crosstalk which dominate on chip ISI, the model is based on a circuit with resistive and capacitive elements. The circuit was leveraged from the work of Dally [2] which explains that it

provides an accurate model of on chip crosstalk. Figure 7 shows the circuit topology from which the initial channel transfer function was modeled.

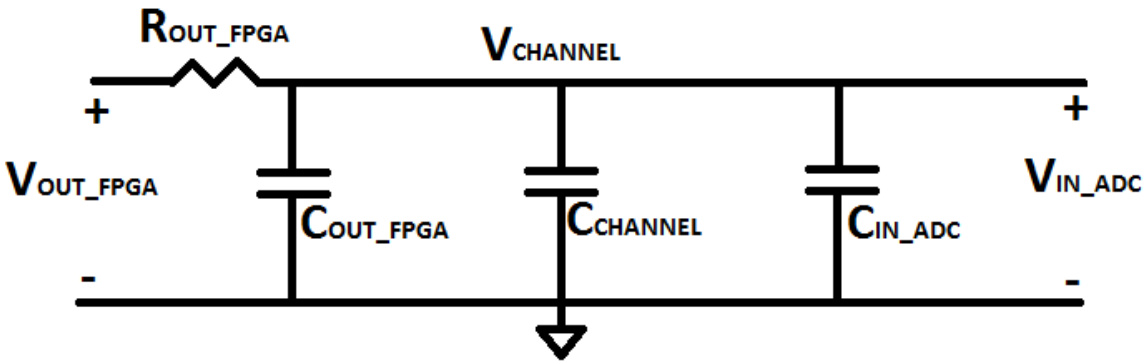

Figure 7: Channel Model

Using basic circuit analysis techniques the model is reduced to a single source resistance driving a capacitive load. The transfer function $H(s)$ is represented by the following equations.

$$H(s) = \frac{V_{in\_adc}(s)}{V_{out\_fpga}(s)} = \frac{1}{R_O C_S + 1}$$

$$R_O = R_{OUT-FPGA}$$

$$C_S = C_{out\_fpga} + C_{channel} + C_{in\_adc}$$

The "transfer function" element in Simulink used this equation to model the channel implemented in hardware. Datasheet values and board schematics were used to determine the values of the resistive and capacitive elements that implement the channel in hardware. A discussion of the actual values used in the hardware model is provided in Section 3.2. Initially, when only the Simulink model was available, the software "Scope" shown in Figure 6 was used to estimate the effect of ISI on the signal in the channel. Using this visual method and an iterative process of refining the transfer function, bit

errors were induced in the Simulink model without actually determining the physical hardware values. The red colored block in Figure 6 displays the channel transfer function model in Simulink.

The next step in the software model development was to create the entire backend including the receiver and equalization architecture. The signal output of the channel was split to model and test the receiver, both with and without equalization, in parallel. The receiver without equalization consisted of a comparator that measured the analog voltage on the receiver front-end and compared it to a reference voltage. The reference voltage was defined in Simulink as 1.65 which is ½ of 3.3. This modeled the receiver front-end in the FPGA hardware and compared any signal less than or equal to 1.65 as a received 0 and any signal greater than 1.65 as a 1. The rectangular, cyan colored block in Figure 6 models the receiver without equalization. Note that the signal output of the receiver was left in the digital realm and represented only as a binary 0 or 1.

The other receiver implementation used a DFE to determine the digital value of the analog received signal. The DFE required more than just a single comparison of an analog value to operate properly, thus a sample and hold module was used in the Simulink model to sample the analog "voltage" once per clock period and deliver that value to the DFE. The sample and hold module is shown as the blue colored block in Figure 6 above and models the ADC in the hardware model. The sampled analog value was used by the DFE design to compute a digital value based off of N previous decisions, where N is the tap length. This design used a 2 tap DFE and is shown as the yellow colored blocks in Figure 6.

The final modules designed in the software model were those required to determine if there were bit errors in the received binary message. This was determined by comparing the received values, both with and without equalization, to the "golden" data generated by the LFSR test pattern generator. When accounting for the appropriate delays in the Simulink model, values from the 20-bit LFSR were taken and an XOR operation was performed between the golden data and the equalized and normally received values. If the values of the golden and received signals did not match then a 1 would be output from the XOR gate which indicated a bit error had occurred. This error checking circuitry, which resides in the FPGA in hardware, is shown as the magenta colored blocks in Figure 6.

Initially the software model ran for a relatively short period of time which simulated 32 transmitted bits to ensure that the DFE implementation was correct. After the model was tuned and ensured to be correct the bit error data was logged to separate simulation data files. A Matlab program was developed to analyze the data and detect the amount of bit errors for BER calculation. This method of simulation allowed for much longer simulations to be run since the data did not need to be visualized with the software scope. All of the blocks that do not have a specific background color in Figure 6 are Simulink specific blocks used to cast data types, resize arrays or add required, underlying functionality to other blocks in the system. The simulated system was designed to model the hardware system as accurately as possible and to provide a means for experimentation and validation once the hardware model was implemented. The results of the Simulink software model are discussed with the hardware model results in Chapter 4.

3.2 HARDWARE

Similar to the development of a software based simulation model, an accurate, flexible hardware model must be designed carefully to focus on the parameters of interest in the system. The hardware system that modeled a high speed serial data link through a channel inducing ISI was designed after the software model was vetted. Initially, the base platform and system components were known before the system was modeled and simulated in Simulink. However, the specific implementation of the hardware and software components was still undecided. A Xilinx Spartan 3 FPGA was used to prototype the hardware design using the Verilog Hardware Description Language (HDL). The TLL 5000 electronic system design platform from The Learning Labs, Inc. had this FPGA integrated into a system reference board.

In addition to the base board, the TLL 6219 ARM9 mezzanine board was attached to the TLL 5000 to provide an interface to the FPGA via an application and device driver written in C. These software components ran on an open source embedded Linux operating system called Micro Monitor (uMon). There was also a general purpose input-output (GPIO) header on the TLL 5000 board which connected to the FPGA IO pins. The header was used to route the signal from the TLL 5000 base board to a breadboard implementing the channel and ADC. Other than the initial hardware and software design components mentioned above, all other design decisions were left open including the implementation of the software, HDL, ADC and channel. Figure 8 shows a visual depiction of the TLL 5000 baseboard and Figure 9 shows the TLL 6219 ARM9 mezzanine board.

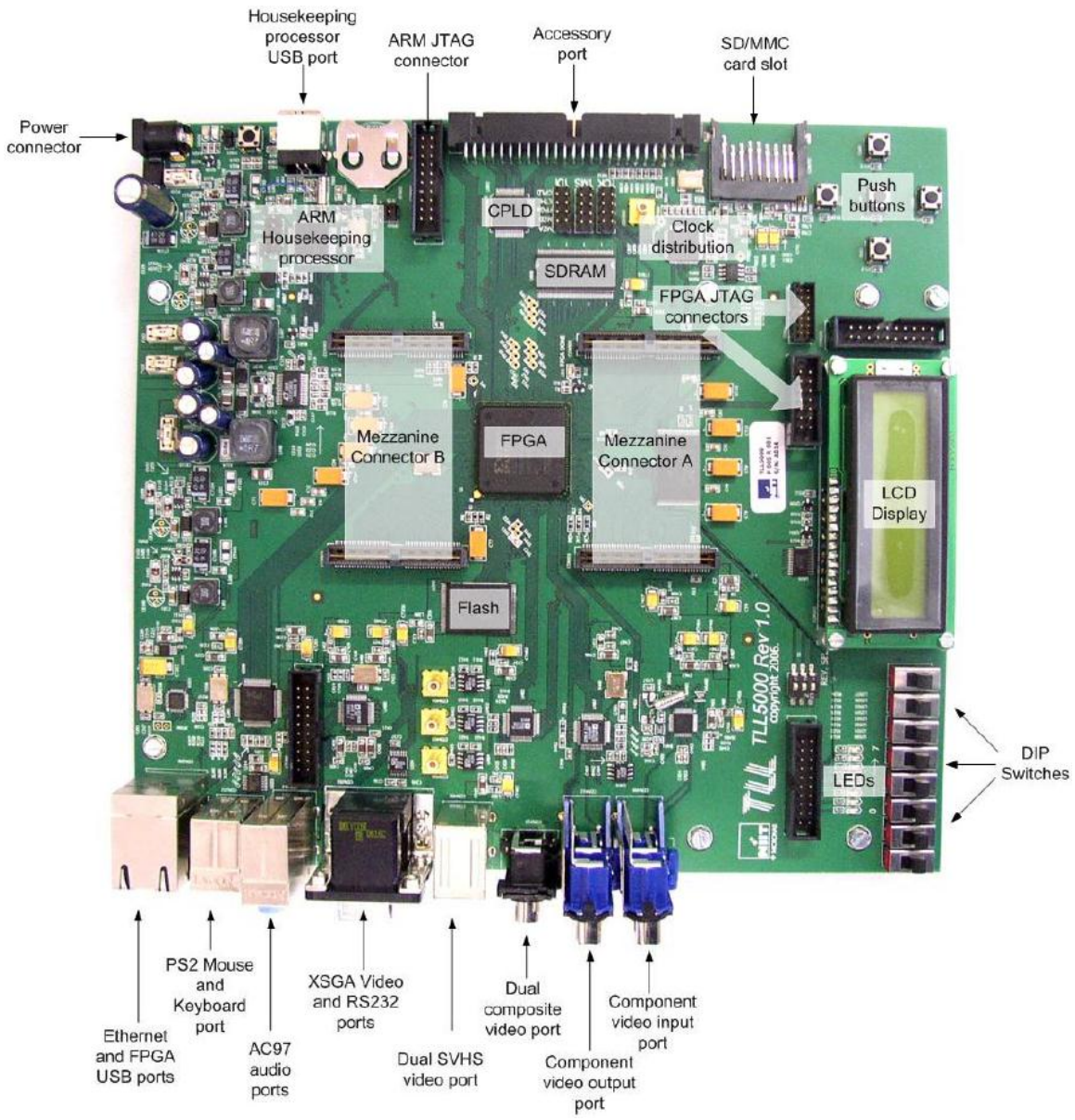

Figure 8: TLL 5000 Baseboard with Xilinx Spartan 3 FPGA

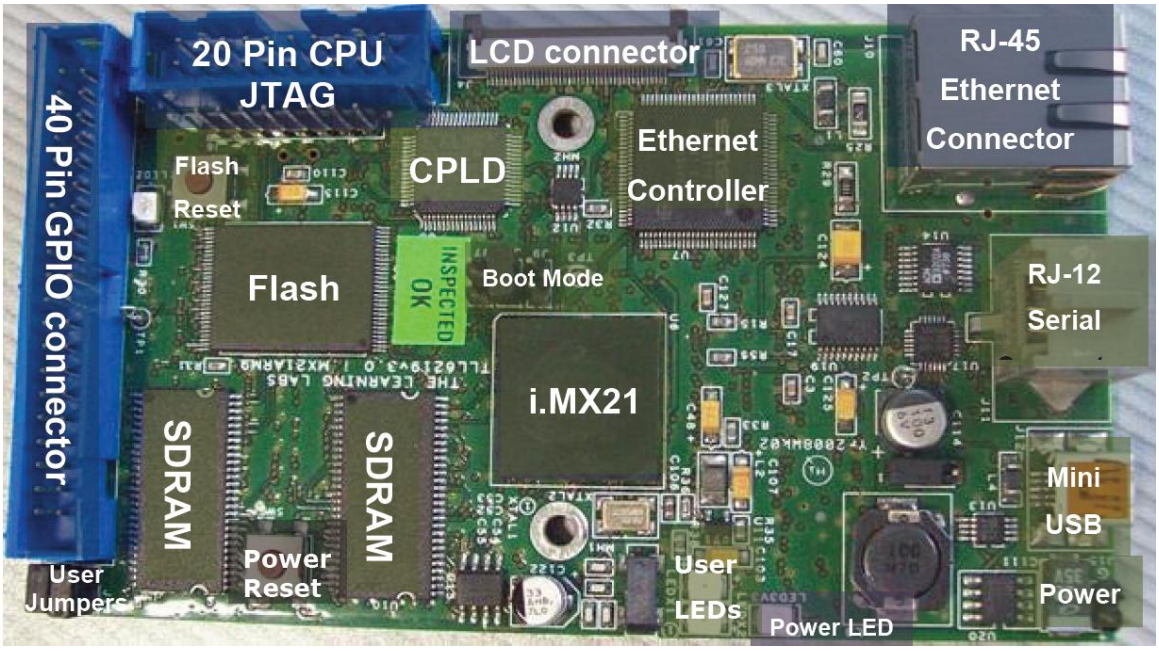

Figure 9: TLL 6219 ARM9 mezzanine board

The goal of the experiments were to focus on hardware implementation, thus the main system functionality was implemented on the FPGA and breadboard. However, software needed to be developed to get data to and from the FPGA for test verification and experimental data collection. Using the C programming language, an application was developed to communicate with the FPGA. The low-level hardware interactions were abstracted from the application by the HAL. Also written in C, the HAL was implemented by a device driver that was registered with the Linux kernel before the application runtime. The application opened the device driver as a file and performed read operations to access FPGA memory which was mapped into the i.MX21 system-on-chip (SoC) bus interface. The device driver accepted an argument from the application that indicated a memory offset address to read from. It then used this offset to put an address out on the system bus and receive data from the FPGA. The HAL then copied the 32-bit data back to the application so it could be displayed on the terminal when

executed. Once the software implementation of communicating with the FPGA was verified and complete, focus was moved towards the actual high speed serial link design and implementation on the FPGA and breadboard prototyping systems.

Similar to the Simulink software model, the first module implemented and tested on the FPGA was the LFSR. As mentioned previously, Verilog was the HDL used to implement the design. The design was synthesized using the Xilinx ISE software package and targeted for the Spartan 3 FPGA device. Using Xilinx iMPACT the *.bit file generated from synthesis was programmed into the FPGA, enabling the HDL implementation of the LFSR to run in hardware. The FPGA main system clock was set to run at 64MHz and a state machine was used to divide the LFSR data bit output down to 1MHz for bandwidth considerations explained later in this section. The output of the 0^{th} order tap of the LFSR was wired to a pin with a Low Voltage TTL (LVTLL) general purpose 3.3V output driver. The analog transmitter circuitry was already implemented on the FPGA and complied with the JESD8C standard, thus this part of the design was not specifically customized. The output of the PRBS signal from the FPGA was connected to a GPIO pin on the TLL 5000 base board header. This signal was then probed with an oscilloscope to verify the correct bit sequence output. Since the LFSR was initialized with the same seed and implemented the same characteristic polynomial as the Simulink model, the bit sequences could be compared for correctness. Thus, the known bit sequence and transmitter modules were verified in hardware before the rest of the system was designed.

Before the channel and ADC hardware were implemented, the most basic loopback setup was tested by connecting the transmitted PRBS signal back to a receiver

pin on the FPGA to determine if the golden signal could be recovered completely without any additional ISI induced upon the channel. This was accomplished using 26AWG wires that routed the transmitted signal from the TLL 5000 base board GPIO header to a breadboard, then back to another pin on the header which connected to an FPGA receiver pin. Similar to the transmitter, the FPGA analog receiver front end was already implemented in hardware and complied with the LVTTL standard. A register within the FPGA was used to logically XOR the received and transmitted PRBS signals to ensure that they were equivalent. The entire signal was able to be received properly when a simple wire was used as the channel. This was the expected result and provided an error free baseline for the hardware setup since there were not any bit errors in the most simple loopback case. The next step in the system development was to induce ISI upon the signal to determine if bit errors could be induced.

Given the 1MHz frequency of the transmitted signal, the *RC* time constant τ is related to the cutoff frequency f_c of the signal by the following equations.

$$\tau = RC = \frac{1}{2\pi f_c}$$

$$f_c = \frac{1}{2\pi RC} = \frac{1}{2\pi\tau}$$

$$C = \frac{1}{2\pi R f_c}$$

The source resistance of the transmitted signal was modeled as a 220Ω series resistor on the TLL 5000 base board. Thus, a value for the capacitance could be determined to provide the desired effect on the PRBS signal. The shortest pulse width for the transmitted signal occurred at 1us for a single 1 surrounded by 0s. However, the

LFSR also has much longer strings of 1s transmitted which increased the pulse width of the signal and thus lowers the energy content from higher to lower frequencies. A capacitor with a value of 5nF placed between the transmitted signal and ground resulted in a cutoff frequency f_c = 145 kHz and was the first time bit errors were observed by transmitting the signal through the RC channel coupled to a 0V ground reference. As the capacitance was increased (maintaining the constant resistance on the TLL 5000 baseboard) in 1nF steps the bit errors also increased as expected. The hardware system was now comprised of a test pattern generator, transmitter, channel inducing ISI, receiver and bit error detector. Figures 10 and 11 show images of an oscilloscope display that had probed the transmitted signal with the simple wire model and also with the channel inducing ISI, respectively.

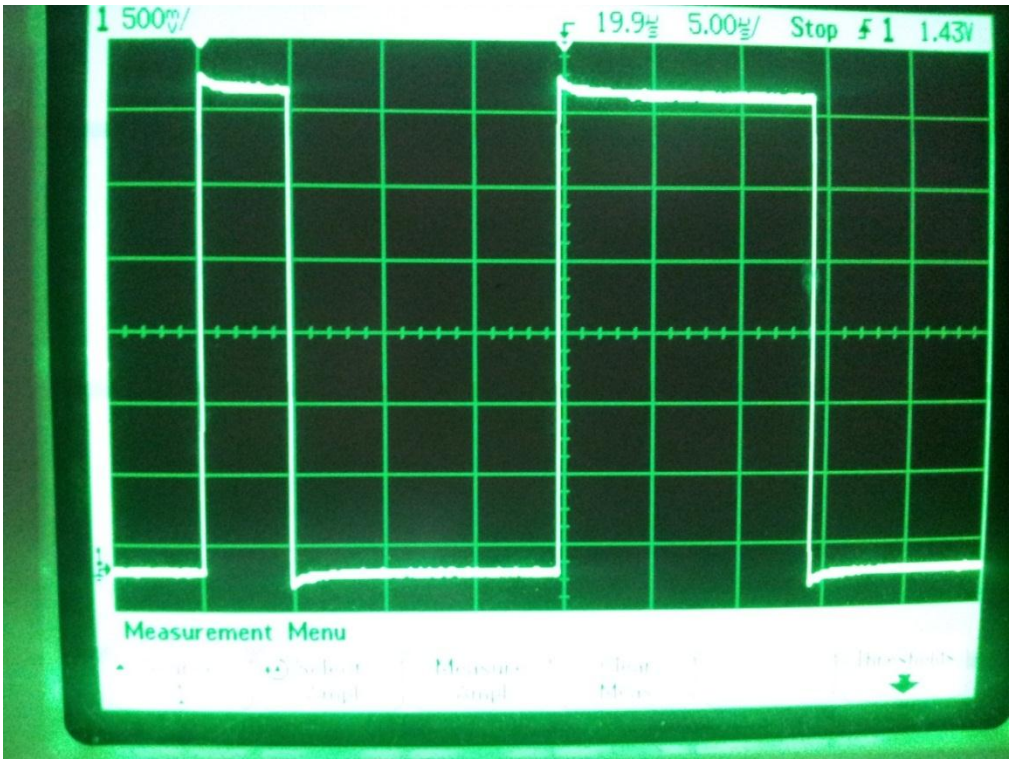

Figure 10: Transmitted PRBS Signal without ISI

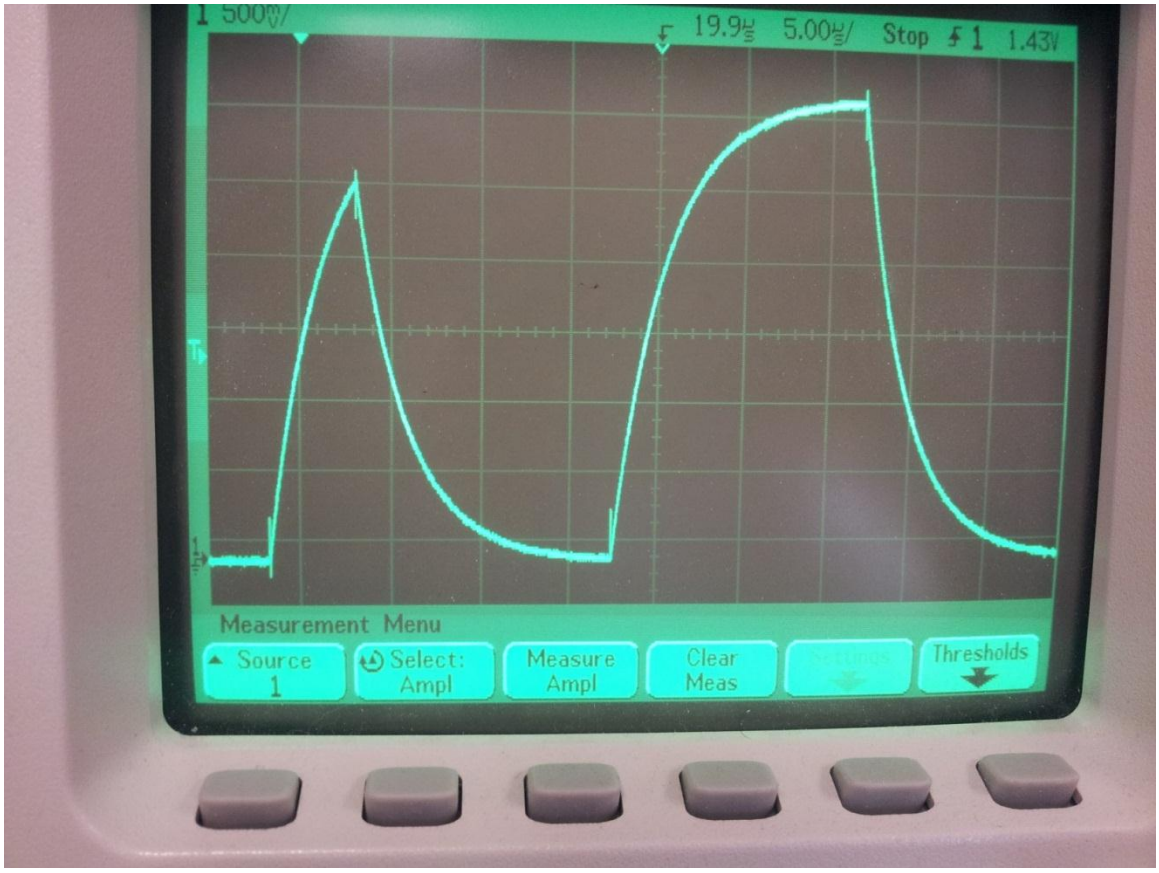

Figure 11: Transmitted PRBS Signal with ISI

A complete setup for the high speed serial data link without equalization had been realized at this point. The next step was to implement the ADC and DFE. Since there were no available data converters integrated into the FPGA or TLL 5000 baseboard, an external ADC was selected for implementation and resided on the breadboard for prototyping. The AD7822 is an 8-bit, 1-channel, 2MSPS ADC from Analog Devices, Inc. and was used to sample the transmitted signal propagating through the channel to provide values for the DFE implementation on the FPGA. The ADC was relatively straightforward to interface with as it did not require a clock but only two sideband control signals from the FPGA, analog reference voltages constructed on the breadboard

and the signal of interest to sample from the FPGA and channel. The output was an 8-bit parallel signal which was routed back to the GPIO port on the FPGA as received signals. The 8-bit ADC dictated the overall clock frequency of the system due to its maximum sample rate. The 1MHz clock rate was chosen to provide adequate time for the ADC to track and hold the analog signal given its specification. The ADC output was sampled on each LFSR data bit edge and stored into a register as the analog sampled value used as the input to the DFE. Figure 12 shows the components of the system modeled and implemented on the breadboard.

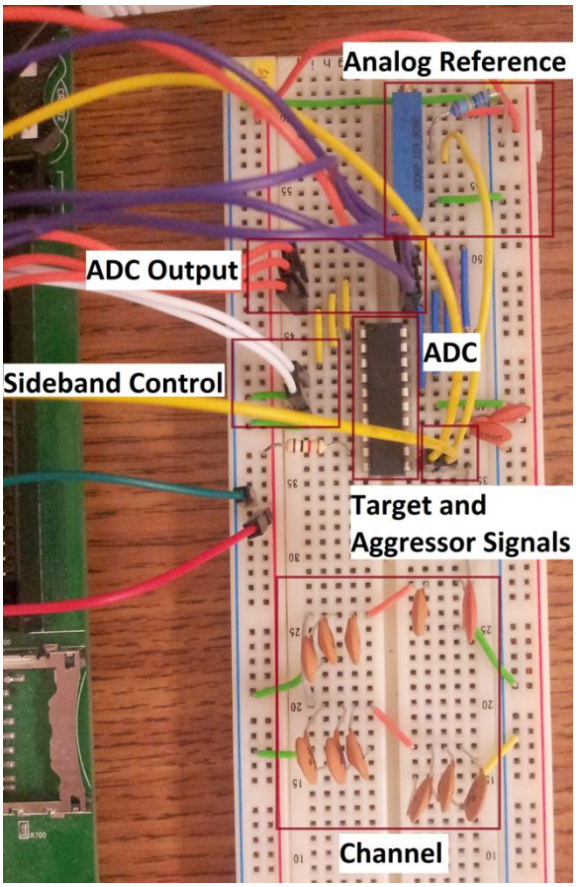

Figure 12: Hardware Modeling System Prototyped on Breadboard

The DFE was implemented on the FPGA using four registers, a difference block, an addition block and a set of shifters to perform the tap scaling. Using the same 5nF capacitance that induced bit errors on the received signal without equalization, the DFE implementation with tap weights of $[C_0, C_1] = [0.25, 0.125]$ was able to recover the signal without any bit errors. Several iterations of adding 1nF of capacitance to the channel continued and the received signal with DFE was able to consistently recover the golden PRBS while the receiver with no equalization reported increasing bit errors. Eventually, with enough ISI induced upon the signal, the receiver with the DFE implementation finally reported bit errors well. At this point the entire hardware model had been setup to match the Simulink software model as closely as possible and the DFE implementation was successful in equalizing the received signal to correct bit errors that would normally occur in a receiver without equalization.

To explore the effect of various types of ISI induced in the channel and DFE tap weight configurations, the hardware model was modified to perform experiments collecting empirical data for performance analysis of the system. Initially the channel was coupled to a 0V ground reference which models a transmission line that is closely shielded by a ground wire or plane and coupled to it as well. In real world systems it is likely that the high speed serial data link signals would be routed closely to other signals of the same type. Though the high speed signaling standards are usually differential, the effects of ISI on one end of the differential pair must be considered. The worst case crosstalk that a signal can experience comes from a complement of the signal. Thus a complement of the transmitted PRBS signal was generated in the FPGA and routed to the GPIO port on the TLL 5000 base board. This signal was then used as a coupling

reference for the target transmitted signal rather than a ground reference. Figure 13 shows a model of the channel with this type of worst case ISI.

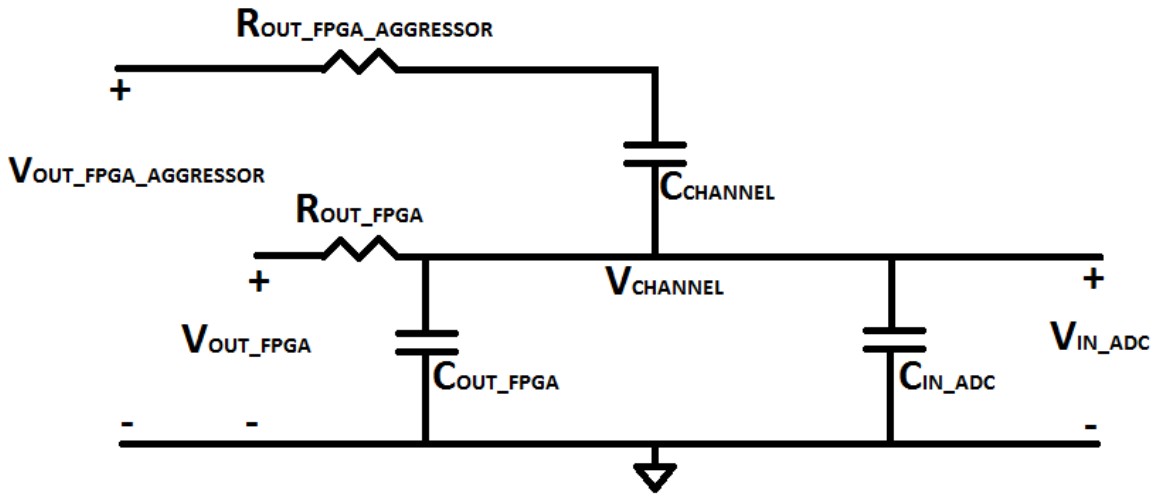

Figure 13: Worst Case ISI Aggressor Signal Model

The equation below provides a model for the transient voltage induced on the target signal $V_{CHANNEL}$ given that a step function is produced on the aggressor signal.

$$V_{CHANNEL}(s) = V_{OUT\_FPGA\_AGGRESSOR}\left(\frac{(R_{OUT\_FPGA} \mid R_{OUT\_FPGA\_AGGRESSOR})C_{CHANNEL}s}{(R_{OUT\_FPGA} \mid R_{OUT\_FPGA\_AGGRESSOR})(C_{CHANNEL} + C_{OUT\_FPGA} + C_{IN\_ADC})s + 1}\right)$$

From this equation it is shown that if the aggressor channel is switching as the complement of the target channel a voltage transient would be induced in the opposite polarity of the target signal. This phenomenon induces more bit errors on the target high speed signal per the same capacitance when compared to a ground reference. The final ISI experiment routed the complement of the 11^{th} tap of the LFSR to the channel. This signal was then used as a coupling reference to see how the target signal would perform

when coupled to a pseudo-random aggressor signal. The results of these experiments are discussed in Chapter 4.

The DFE tap weights were another set of parameters evaluated to assess the equalization performance. For a given bit transmitted in period *N*, the tap weights are ideally set to the level of the signal when it has dispersed into bit periods *N+1* and *N+2* for the first and second order tap weights, respectively. This allows the signal contribution of previous bits to be subtracted from the currently received bit. Since the transmitted signal was pseudo-random and the amount of ISI induced upon the signal was varied, it would require much more complicated circuitry to adjust the tap weights on the fly. Thus, to understand the effect that tap weight values had on the BER for the various ISI sources, a set of values were selected for use with each sweep of channel ISI. The tap constants were implemented as shifters and adders in hardware. For example, to implement a tap constant of 0.375 the following operation was performed:

$$S_{WEIGHTED} = (\text{S} >> 2) + (\text{S} >> 3)$$

It can be seen that the received signal r_n can be described in discrete time as a sum of constants at each time interval.

$$r_n = a_n h_0 + (a_{n-1} h_1 + a_{n-2} h_2) + \sum_{i, i \neq 0,1,2} a_{n-i} h_i$$

$$= \text{Signal} + \text{Dominant ISI} + \text{Residual ISI}$$

The primary signal information is received when $i = 0$, followed by the dominant ISI terms when $i = 1, 2$ and finally the residual ISI when $i > 2$. For a transmitted pulse

signal the received signal should decay with time exponentially. Intuitively each successive a_n term should be smaller than the previous one, but the specific values depend on how much charge has built up on the transmission line. Thus, using the software model and empirical testing, four sets of tap constants were chosen for the experiments:

$$S_0 = [0.375, 0.125]$$

$$S_1 = [0.25, 0.\ 25]$$

$$S_2 = [0.25, 0.125]$$

$$S_3 = [0.125, 0.125]$$

The selection of these weights covers a reasonable range of values that a bit would likely have when dispersed into the *N*+1 and *N*+2 bit periods. The tap weights were the final parameter varied in the hardware experiment.

The model and experiment setup described in this chapter provide an explanation of the specific implementation of the high speed serial link system under test. It also described how the system parameters of interest were design to be isolated for performance evaluation. Both the hardware and software models complemented each other throughout the design and development process. The models also provided verification references for results that were seen in the other models. The following chapter will discuss the results of the experimental data in more detail.

Chapter 4:

Experimental Findings

The simulation and hardware experiments described in Chapter 3 were performed to explore the effects of ISI on high speed serial links and how adaptive equalization hardware can be used to reduce the channel effects on the signal. Empirical data was gathered from the experiments to evaluate the modeling and implementation of the systems. This chapter presents the results of the experimental data and provides commentary on the findings to highlight the trends and major takeaways that can be applied to the goals set out in this report.

4.1 RESULTS

Three distinct types of ISI were induced upon the target high speed serial link signal. This was accomplished by varying the type of reference (aggressor) signals that the target signal was coupled to through capacitance. The three flavors of aggressor signals were: 0V ground, complement of the target PRBS signal, and an arbitrary PRBS reference. Within each of these separate ISI inducing sources the discrete capacitance used to couple the target and aggressor signals was also varied from 0nF to 20nF in 1nF resolutions. The coupling capacitance also includes the transmitter, ADC and receiver pin capacitances as well. These capacitances remained static throughout the experiments. Finally, the DFE tap weights were also varied to investigate their effect on the performance of the equalization circuitry. Figures 14-16 show the results from the software simulations that used the same transfer function values as the hardware experiment. Figures 17-19 show the results from the actual hardware experiment per all of the parameters.

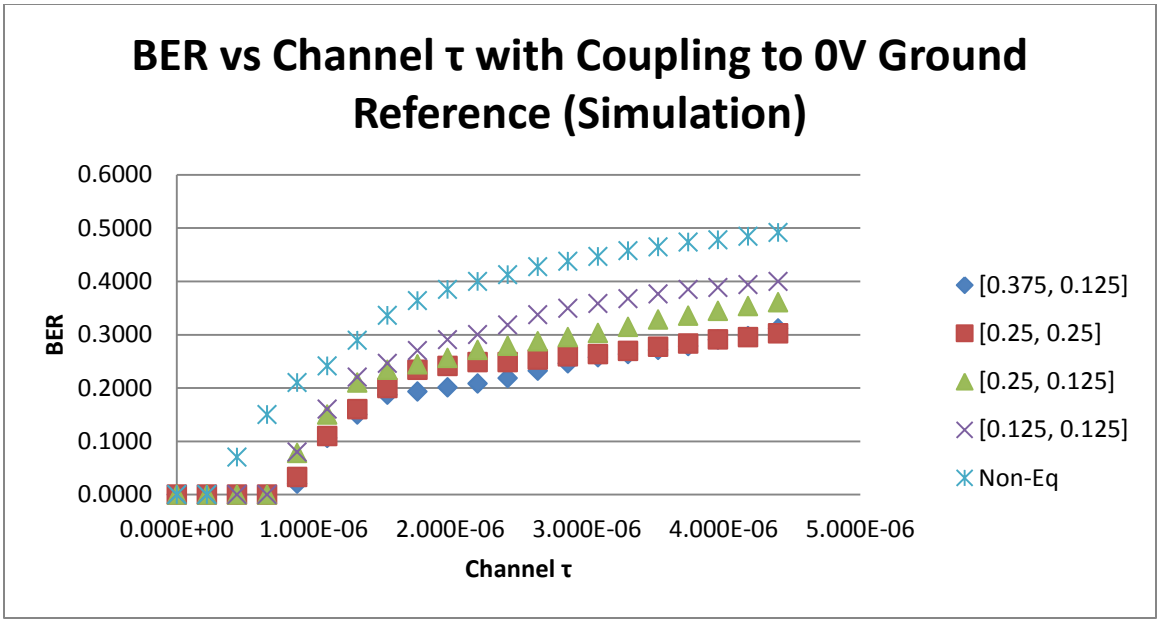

Figure 14: BER versus Channel τ with Coupling to 0V Ground Reference (Simulation)

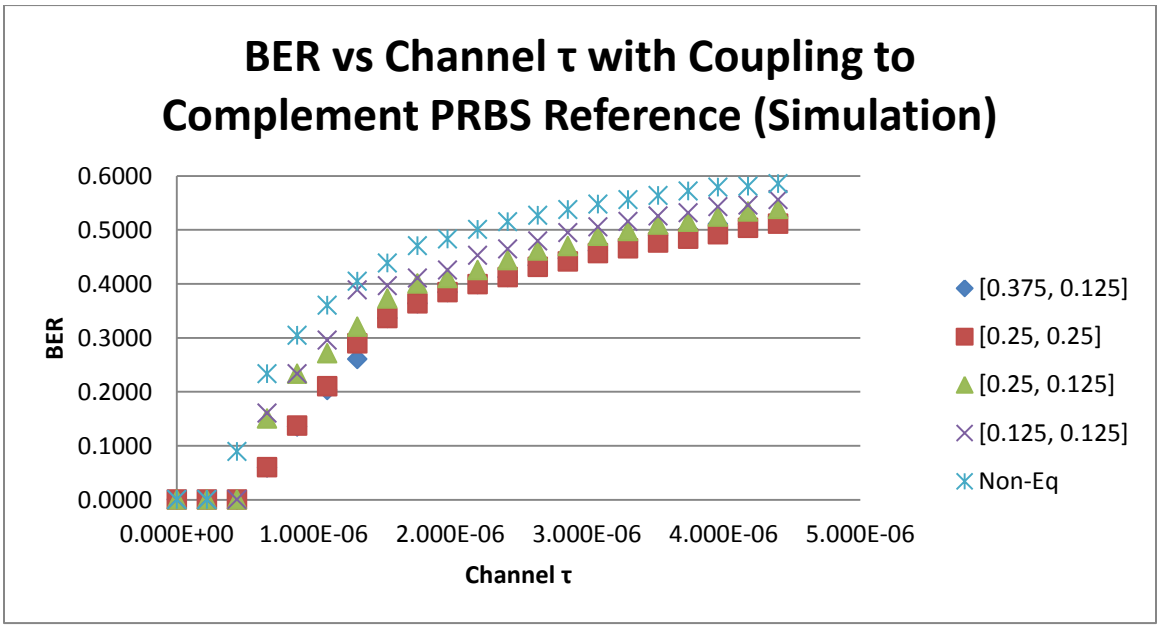

Figure 15: BER versus Channel τ with Coupling to Complement PRBS Reference (Simulation)

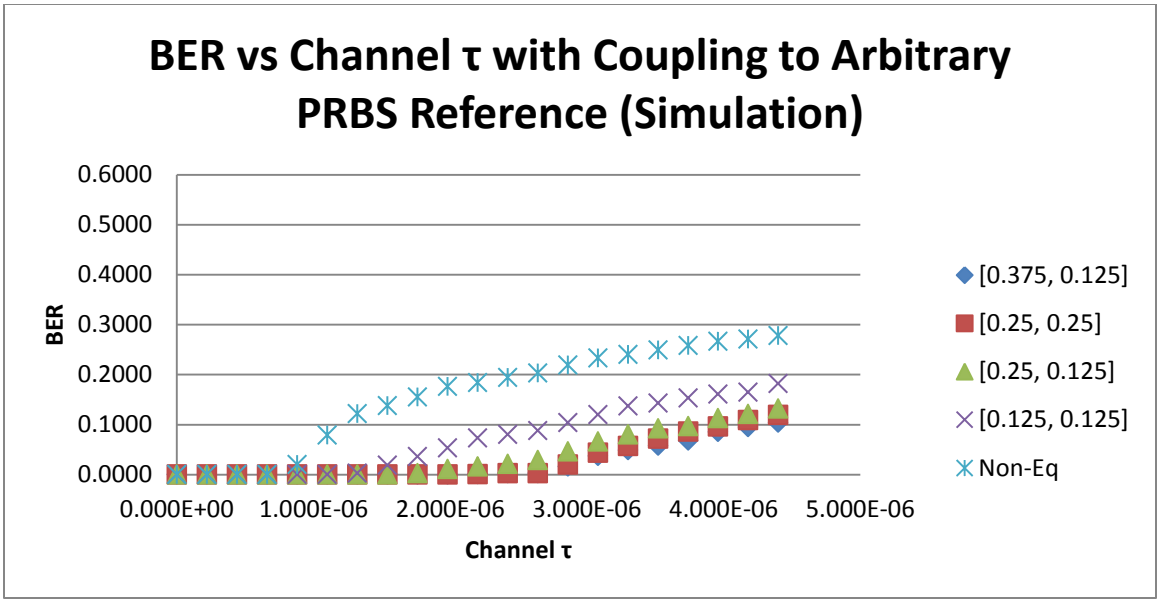

Figure 16: BER versus Channel τ with Coupling to Arbitrary PRBS Reference (Simulation)

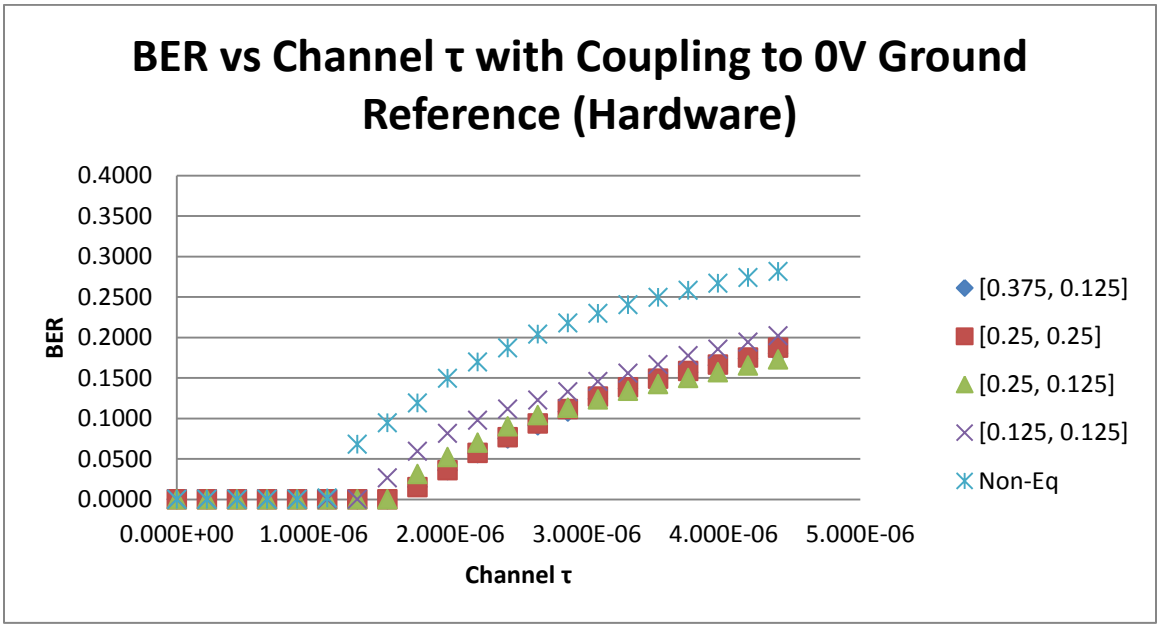

Figure 17: BER versus Channel τ with Coupling to 0V Ground Reference (Hardware)

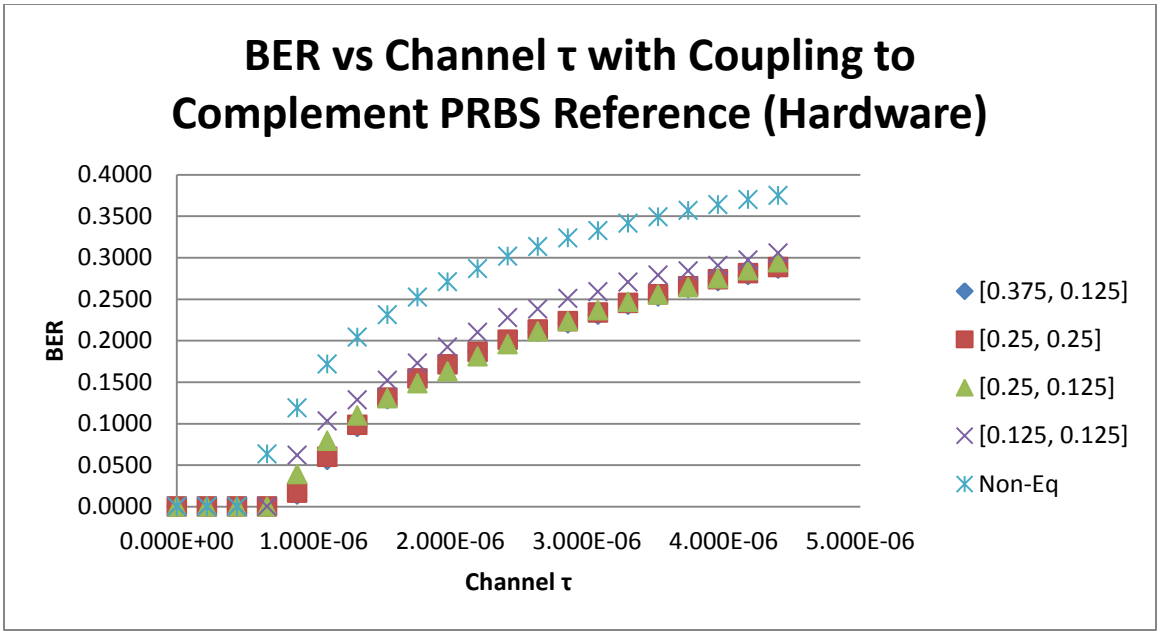

Figure 18: BER versus Channel τ with Coupling to Complement PRBS Reference (Hardware)

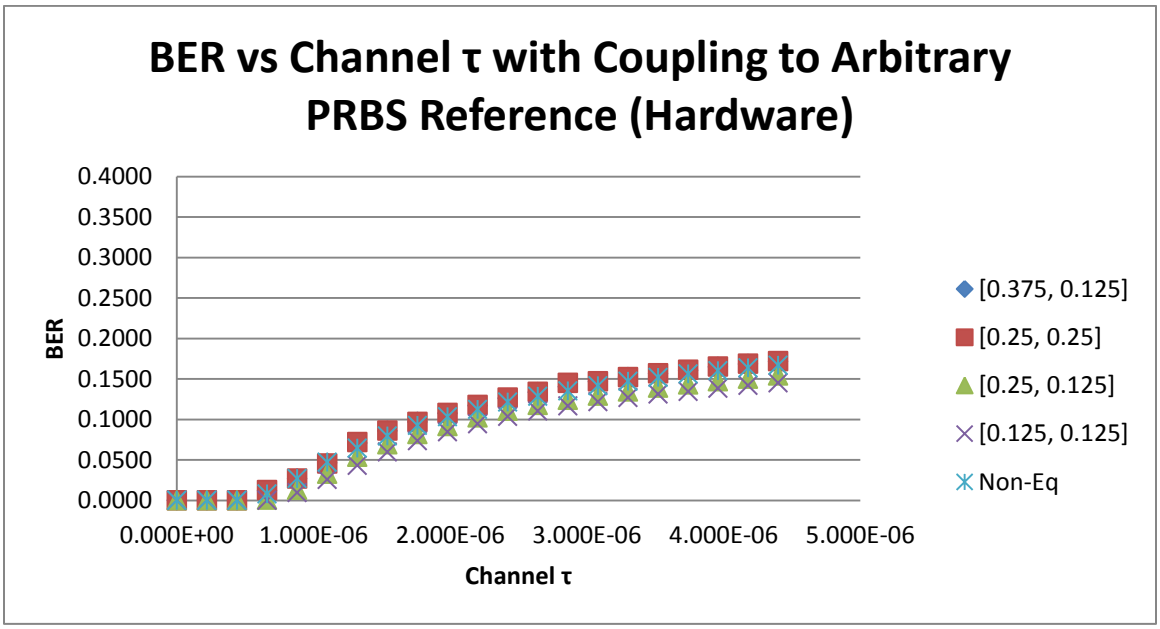

Figure 19: BER versus Channel τ with Coupling to Arbitrary PRBS Reference (Hardware)

4.2 DISCUSSION

The results of the simulation and hardware experiments produced data that correlates common trends in both of the models. The first observation to note is that all of the DFE implementations are able to provide sufficient receiver equalization for at least one data point of ISI that corrects all bit errors where the receiver without equalization has bit errors. This proves that the DFE implementation is effective in both the hardware and software model. Also, the general piecewise function of the BER across the channel transfer function τ is similar between the two models. At first a linear BER at or near 0 occurs for relatively low amounts of ISI. As the coupling increases, bit errors begin to occur and cause the BER curve to grow in an exponential fashion with increasing τ. Another similarity between the hardware and simulation model is that they both project that the worst case coupling source is the complement PRBS reference, followed by the 0V ground reference, and finally the arbitrary PRBS reference induces the lowest amount of bit errors relative to the others.

Given the common data trends between the hardware and simulation models of the high speed serial link system with DFE, a general understanding of the effects of different types of ISI sources and the performance of equalization circuitry can be realized. However, there are differences between the hardware and simulation results as well. The simulation model always produced a worse BER for the receiver without equalization when compared to the hardware model with the same type of ISI reference source. The BER values are also higher in the simulation model for the receivers with DFE when the 0V ground or complement PRBS references are used as ISI sources. However, the BER values are less in the simulation model when the arbitrary PRBS reference is used to induce ISI.

There could be several reasons for differences in these results. It is difficult to know exactly when the FPGA and ADC receivers exactly sample the analog voltage value for comparison, thus the sample timing between the models likely had some amount of timing error between them. Variations in the hardware sampling time would have made it nearly impossible to correctly time the simulation model even if the specific analog track and hold time was known. This timing mismatch certainly has an impact on the receiver's ability to correctly interpret a bit as a logic 0 or 1 value. Another possible source of mismatch between the two models is the precision in which the analog values are measured. In the simulation model the analog values can be represented as a *double* data type whereas the ADC only represents the value as an 8-bit number. Again, the resolution in the analog representation of the signals was likely a source of difference between the two models. Finally, it becomes much more complicated to measure an model all of the physical effects of the real world hardware system in a simulation. The general BER trends align between the two models, empirically proving the channel model is accurate. However, it would be extremely difficult to incorporate all of the higher order effects of the hardware model in the simulation and this differences in the data arise.

The first hardware specific observation to highlight is that at least two particular DFE implementations always receive the transmitted data with fewer bit errors than the non-equalized receiver. When the target signal is coupled to a 0V ground or complement PRBS aggressor signal, all DFE implementations always perform better than the non-equalized receiver. The observations from the hardware experiment confirm the theoretical and simulated results. A receiver with an adaptive DFE can always perform

better than a non-equalized solution in the test environment described in this report. This highlights the achievement of the main goal of the research performed for this report.

Within the empirical data there are further findings that also need to be understood. As anticipated, the BER for all receiver implementations is greater for the case when ISI is induced upon the signal by coupling it to the complement PRBS signal reference. When the target signal is coupled to an arbitrary PRBS reference, the BER is actually lower than the coupling to a 0V ground reference. One can intuitively reason that the general trends of the results are correct. A target signal that is coupled to an aggressor that is switching voltage levels at the same time and in an opposite direction will always have an electric potential difference induced upon it that acts against the direction it is switching. Thus if the signal is attempting to discharge the wire to represent a digital 0 it will have some amount of transient charge added to it which effectively delays the discharge in time. This directly increases the amount of ISI induced upon the signal as it is dispersed wider into other bit periods. When the target signal is coupled to a 0V ground reference it is constantly charging and discharging a capacitance to a static voltage reference, thus the rise and fall times should be quicker than when coupled to an opposite signal. Finally, in the case of an arbitrary PRBS reference both the target and aggressor signals experience times when they are both switching in the same direction and others when they are opposite in direction. This effectively averages out the charging and discharging of the capacitance and results in a BER that is lower than both the 0V ground and complement PRBS references.

The general curve of the BER versus the channel τ for each receiver implementation matches well when compared to the other implementations. However,

there are instances where the curves for a particular DFE receiver type cross in the figures above. Due to the nature of the 2-tap DFE architecture there are two previous decisions removed from the current received bit period. This suggests that the second order tap weight begins to show its ability to more accurately filter out ISI as the coupling increases. For example, consider the experiment where the target signal was coupled to the complement PRBS reference. The [0.25, 0.25] tap weighted DFE implementation performs better than the [0.25, 0.125] tap while the coupling is relatively low. However, when the coupling increases the [0.25, 0.125] tap weight begins to operate with fewer bit errors. This is due to the fact that the transient rise and fall times are more accurately represented by the [0.25, 0.125] tap weights than the [0.25, 0.25] tap weights. Thus, the [0.25, 0.25] weights are overestimating the contribution of the bit received 2 clocks before the current bit period and subtracting it from the current bit period. Ideally the channel could be characterized before actual serial data is transmitted to tune the DFE tap weights and provide the optimal setting for a static channel. Additionally, if the channel is dynamic itself then more complicated circuitry could be used to adapt the tap weights on the fly and keep them tuned to optimal levels for the given channel characteristics at any instant in time.

As mentioned previously the primary form of ISI considered in the experiments was induced from the capacitive channel effects. This represents the dominant form of ISI that is induced on-chip [2]. Since the hardware system was prototyped on a circuit board platform with signal wires connecting to a breadboard, inductive properties of the system could have also been considered as part of the channel model. To empirically eliminate the possibility that mutual inductance had a major effect on the signal propagating through the channel, a side experiment was performed to rule out these

effects. This consisted of using 600mm of wire to connect the signals from the TLL 5000 to the breadboard instead of the typical 150mm used in all of the previous experiments. The results were compared to previous data and there was not a significant amount of difference in bit errors received, i.e., less than 0.1%. Also, the 600mm wire was then twisted and curled upon itself several times in an attempt to blatantly induce mutual inductance given classical methods. Again a set of experiments was performed to compare the results to the baseline, un-twisted 150mm wire and the results were still less than 0.1% in difference. This side experiment effectively ruled out the possibility that mutual inductance had a significant effect upon the modeling or experiments described in this report.

Chapter 5:

Conclusion

Modern computing platforms require high speed serial links to transfer large amounts of data between the various system components for processing. Serial links can operate at much higher clock rates than the antiquated parallel interfaces due to data independence between the links. However, these higher clock rates are subject to channel effects that induce ISI upon the signals and can result in bit errors when the data is received. Receiver equalization is a commonly implemented to overcome the channel effects. Specifically a 2 tap DFE can provide sufficient equalization to enable PCIe 3.0 data rates of up to 8 GT/s. The models, theories and experiments described and implemented in this report sought to explore the effects of ISI upon a signal and the use of a DFE to filter out the unwanted channel effects. Through the experimental data gathered in the simulation and hardware implementation of a high speed serial link system with DFE, it was shown that receiver equalization can filter ISI. In certain models the receiver with DFE was able to correct all bit errors when a receiver without equalization produced a BER of 9.5%. The goal was to implement a DFE in hardware to reduce bit errors for received data sent through a channel with ISI. Through the theory and data presented in this report it is apparent that this goal was achieved.

Though the fundamental goal of this report was met there is certainly room for model improvement, experiment enhancement and additional concepts to be explored in the future. One area for enhancement would be to increase the clock rate of the high speed serial link. The clock rate was limited by the bandwidth of the ADC selected for conversion of the analog signal to a digital value that the DFE input could operate upon.

The 8-bit ADC from Analog Devices, Inc. was selected for its low pin count, straightforward interfacing and low cost. The clock rate was kept on the order of single digit MHz due to the ADC bandwidth limitation. An ADC with a higher bandwidth could be selected to more accurately model the data rates that are present in modern high speed serial link signaling schemes such as PCIe 3.0. Also, if the ADC had a higher resolution than 8-bits the analog signal would be more accurately represented in the digital space and the DFE could have a more accurate input for its equalization implementation.

In the experiments performed for this report only capacitive coupling to another wire was considered. In real world systems, the effects of mutual inductance are more pronounced in the off chip signal routing. To provide a more accurate model for ISI, considerations of mutual inductance could be included in the experiments. Also, the channel effects described in this report were induced by various voltage references coupled with the target signal. In reality, there are many more channel effects that could cause bit errors on the signal. Jitter is one channel effect that causes the periodicity of a signal to deviate from the expected rate [17]. Jitter is a common metric in all systems since it is very difficult to ensure that signals are synchronized to an exact instant in time. Many systems are able to tolerate certain amounts of jitter but as the clock rate increases, any deviation in the periodicity of a signal has a higher percentage effect since the period itself is decreasing.

Another channel effect that often occurs in modern systems is the modulation of other alien signals upon the target signal. Most modern computing platforms have mixed signal capabilities implying that signals of various frequencies and amplitudes are all

routed relatively near each other both on and off chip. These mixed signals can interfere with each other by modulating the information they are carrying to other nearby signals. All of these additional ISI sources could be considered in a more refined model and perhaps mitigated through the addition of supplemental signal conditioning hardware.

The simulation and hardware implementation of the system presented in this report provided a means for modular system components to be added for performance analysis. For instance, the channel was able to be easily modified by adding discrete components on the breadboard and the FPGA allowed for relatively quick development of receiver equalization hardware implementation. This same setup could be used to explore various types of equalization techniques such as a least means squared or recursive least squares filter. Future works could leverage the software and hardware setup to implement more advanced equalization techniques to explore their properties on similar or more advanced channel effects.

Bibliography

[1] G. Forney Jr., "Maximum-likelihood Sequence Estimation of Digital Sequences in the Presence of Intersymbol Interference," *Information Theory, IEEE Transactions on* , vol. 18, no. 3, pp. 363-378, May 1972.

[2] W. J. Dally, *et al*, *Digital Systems Engineering*, Cambridge University Press, 1998.

[3] L. Wang, *et al*, *VLSI Test Principles and Architectures: Design for Testability*, 2006.

[4] C. Shannon, "A Mathematical Theory of Communication," University of Illinois Press, 1949 (reprinted 1998).

[5] PCI-SIG, PCIe® Base 3.0 Specification, www.pcisig.com/specifications/, November 2010.

[6] M. Austin, "Decision-feedback Equalization for Digital Communication Over Dispersive Channels," M.I.T. Res. Lab Electron., Tech. Rep. 461, August 1967.

[7] D. George, *et al*, "An Adaptive Decision Feedback Equalizer," *Communication Technology, IEEE Transactions on*, vol. 19, no. 3, pp. 281-293, June 1971.

[8] H. Nyquist, "Certain Topics in Telegraph Transmission Theory," *Transactions AIEE (Commun. Electron.)*, vol. 47, pp. 617-644, April 1928.

[9] S. Golomb, *Shift Register Sequences*, 1982.

[10] P. Alfke, "Efficient Shift Registers, LFSR Counters, and Long Pseudo-random Sequence Generators," www.xilinx.com/support/documentation/application_notes, July 1996.

[11] B. Widrow, "Thinking About Thinking: The Discovery of The LMS Algorithm," *Signal Processing Magazine, IEEE*, vol. 22, no. 1, pp. 100-106, January 2005.

[12] V. Panuska, "An Adaptive Recursive-least-squares Identification Algorithm," *Adaptive Processes (8th) Decision and Control, 1969 IEEE Symposium on* , vol. 8, pp. 65, November 1969.

[13] A. Fertner, “Improvement of Bit-error-rate in Decision Feedback Equalizer by Preventing Decision-error Propagation,” *Signal Processing, IEEE Transactions on*, vol. 46, no. 7, pp. 1872-1877, July 1998.

[14] C. Belfiore, *et al*, “Decision Feedback Equalization,” *Proceedings of the IEEE*, vol. 67, no. 8, August 1979.

[15] W. J. Dally, *et al*, “Transmitter Equalization for 4Gb/s Signaling,” *IEEE Micro*, pp. 48-56, January/February 1997.

[16] P. Marwedel, *Embedded System Design: Embedded Systems Foundation of Cyber-Physical Systems*, Springer, 2011.

[17] B. Razavi, *Design of Analog CMOS Integrated Circuits*, McGraw-Hill, 2001.